(12) UK Patent Application (19) GB (11) 2 325 118 (13) A

(43) Date of A Publication 11.11.1998

(21) Application No 9805691.4

(22) Date of Filing 17.03.1998

(30) Priority Data
(31) 08826564 (32) 04.04.1997 (33) US

(51) INT CL[6]
H04B 7/26 // H04M 1/72

(52) UK CL (Edition P)
H4L LDLT L1H3

(56) Documents Cited
US 5434905 A

(58) Field of Search
UK CL (Edition P) H4L LDJJ LDLT
INT CL[6] H04B 7/212 7/26 , H04L 7/00 7/02 7/033 7/08 7/10 , H04M 1/72
ONLINE DATABASE: WPI

(71) Applicant(s)
Casio PhoneMate Inc
(Incorporated in USA - California)
20665 Manhattan Place, Torrance, California 90501, United States of America

(72) Inventor(s)
Brady N Guillaume

(74) Agent and/or Address for Service
Marks & Clerk
57-60 Lincoln's Inn Fields, LONDON, WC2A 3LS, United Kingdom

(54) Abstract Title
Cordless telephone synchronisation for high speed high volume data transfer

(57) A cordless telephone base unit (40 fig.2) generates, encodes and transmits a digital data burst having periodic transitions. A cordless handset 18 has a receiver 20 which samples the encoded data burst and a synchronisation device 30-36 for maintaining synchronisation between the receiver and the transmitted encoded data burst in accordance with a characteristic of an expected transition time of the digital data burst. The synchronisation device operates by comparing an actual time of transition to the expected time of transition and adjusting a sampling point of the encoded data burst by a one sided majority vote of samples around the sampling point. Thus minor time discrepancies between transmitter and receiver are compensated. The data is Manchester encoded into bit-pairs '10' or '01' and the receiver samples at four times the transmitted bit rate giving 8 samples '11110000' for a transmitted bit-pair '10'. The 8 samples are fed to synchronisation register 36 and bit slippage is detected if the two centre samples in the synchronisation register 36 match. The one sided majority vote test determines which direction slippage has occurred and the receiver sampling point is adjusted accordingly.

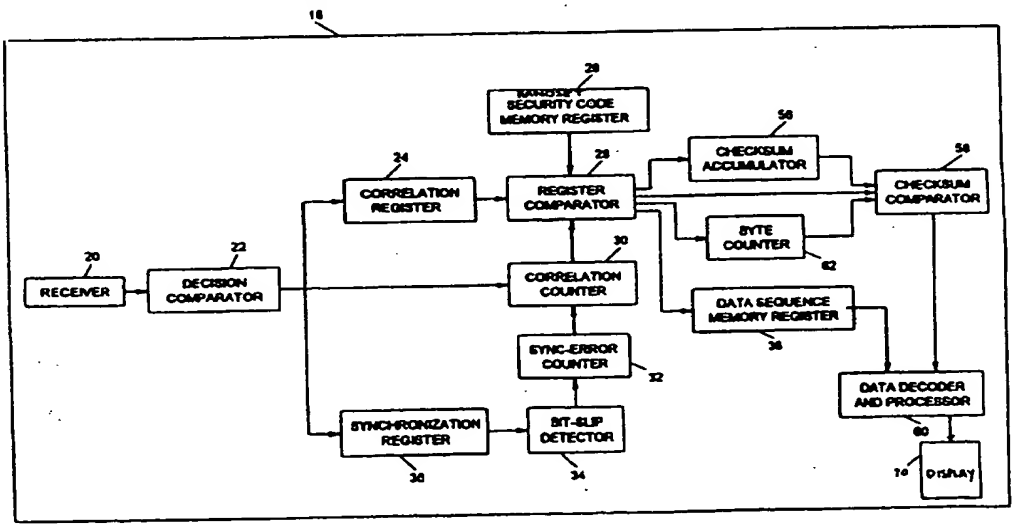

FIG. 3

GB 2 325 118 A

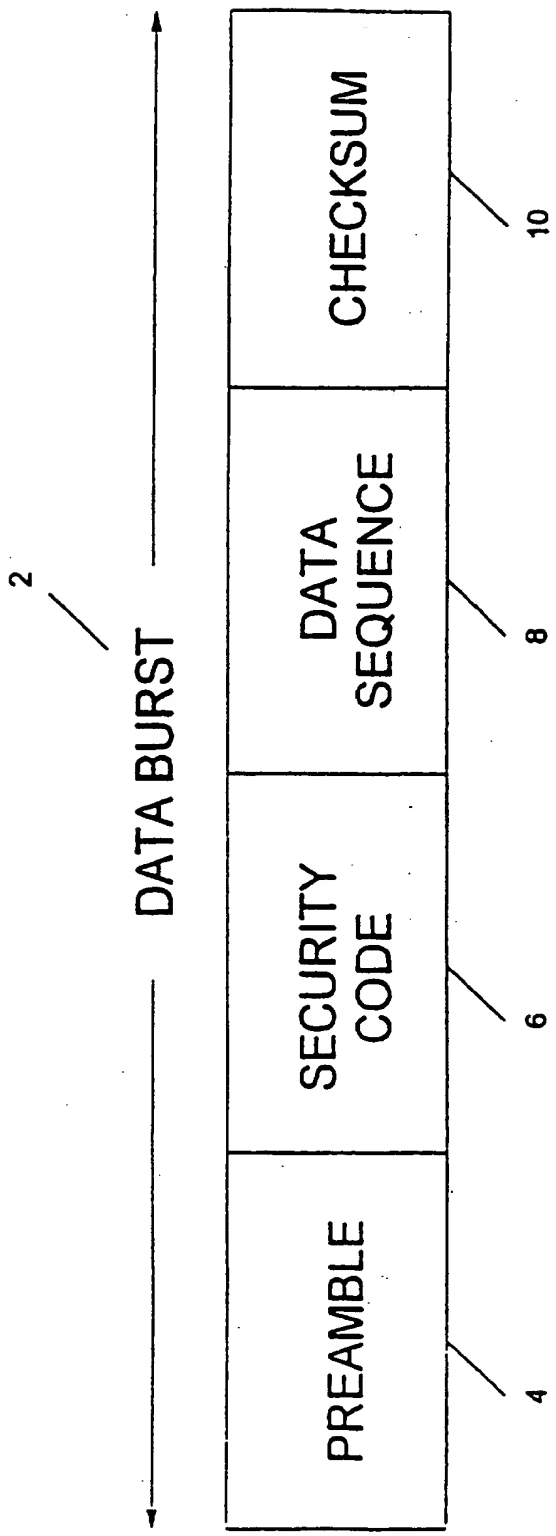

FIG. 1

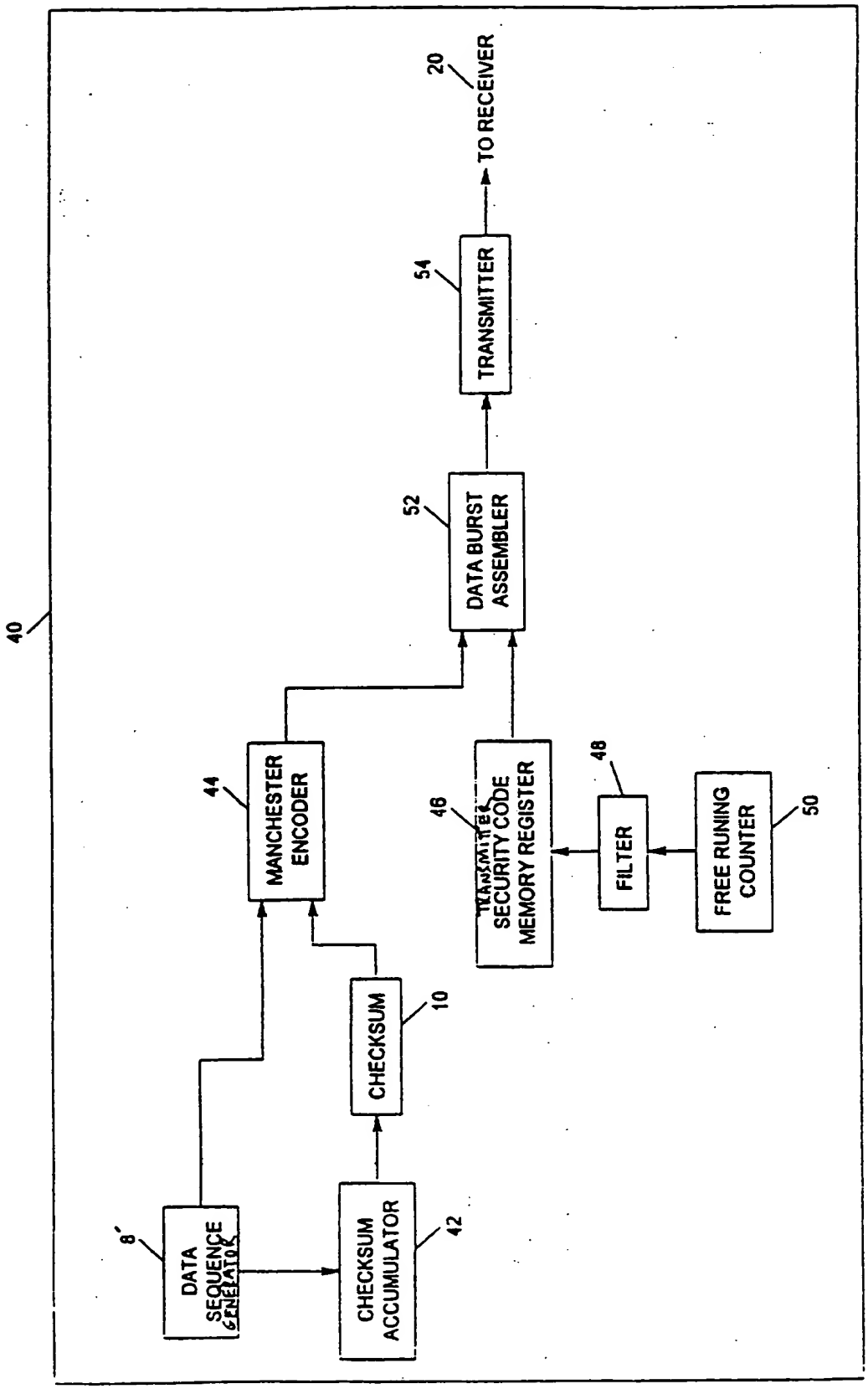

FIG. 2

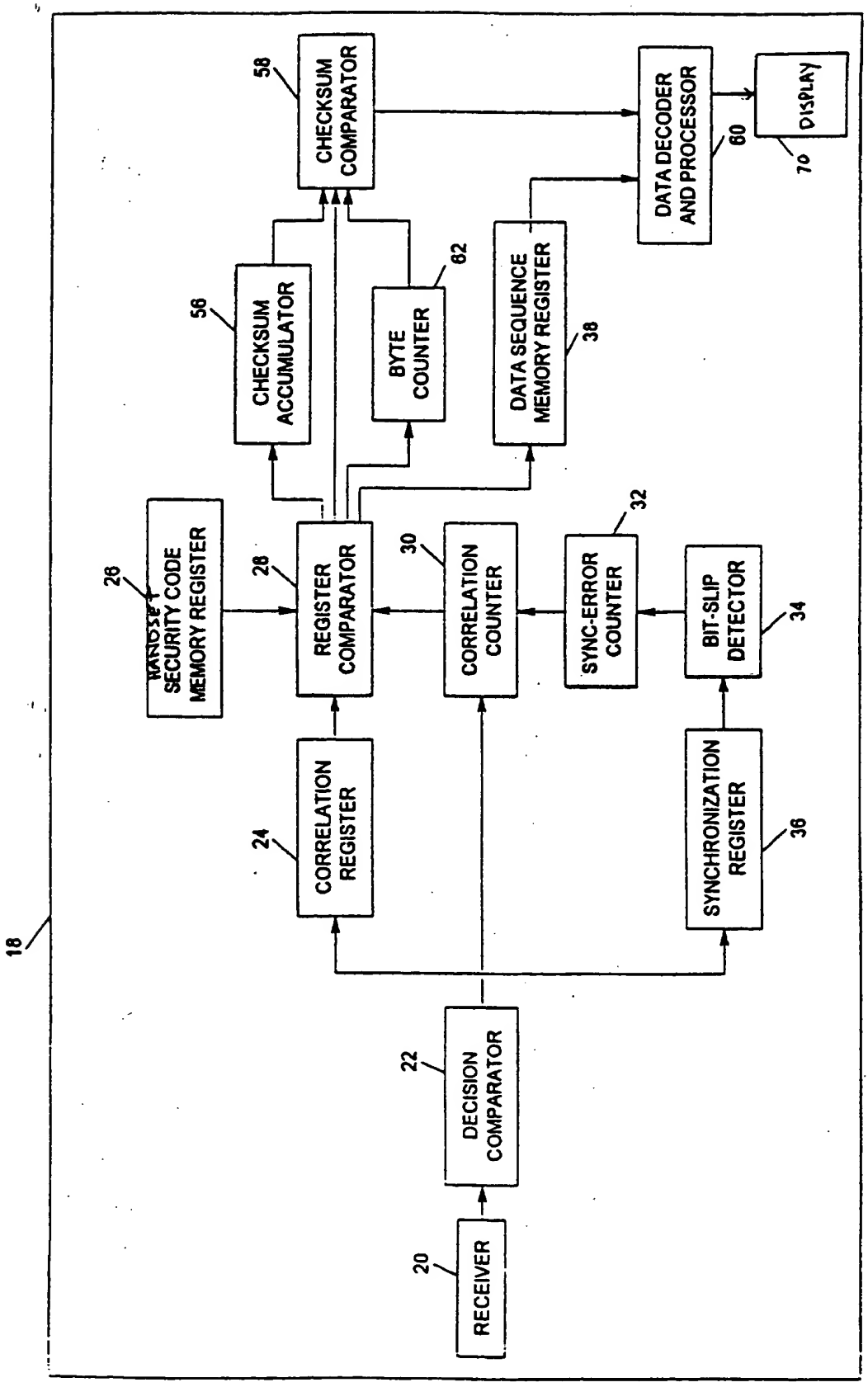

FIG. 3

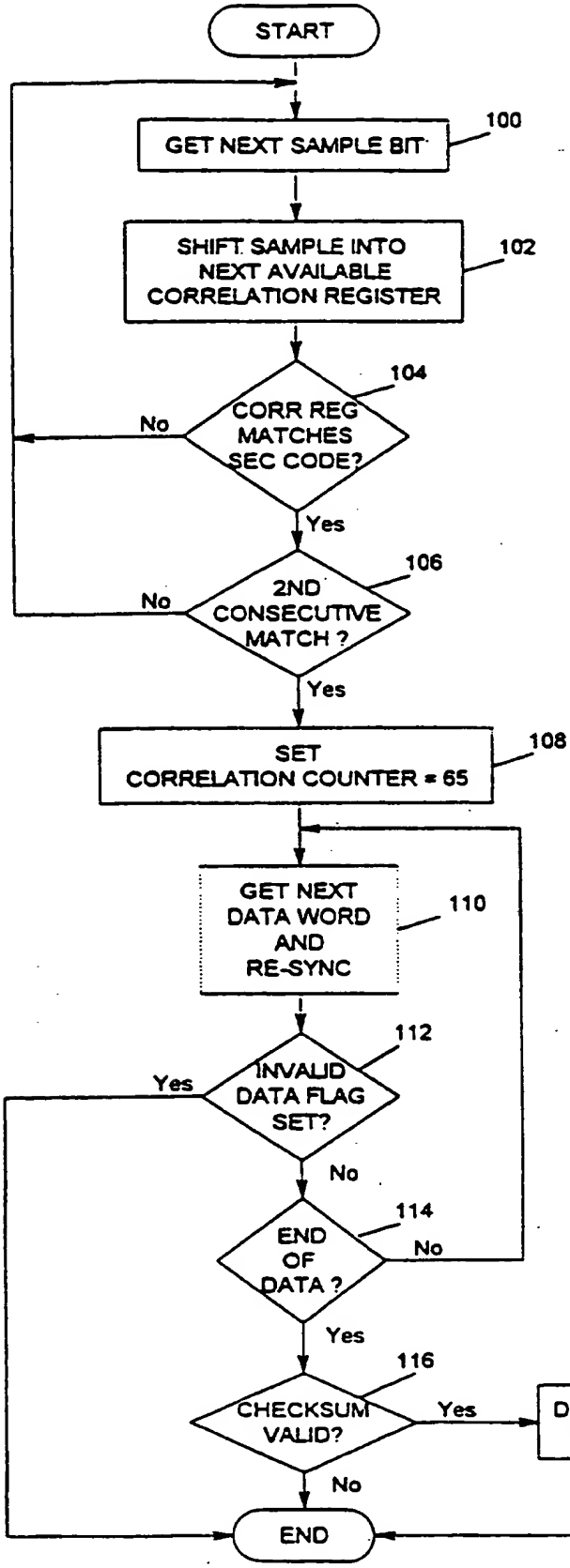

FIG. 4

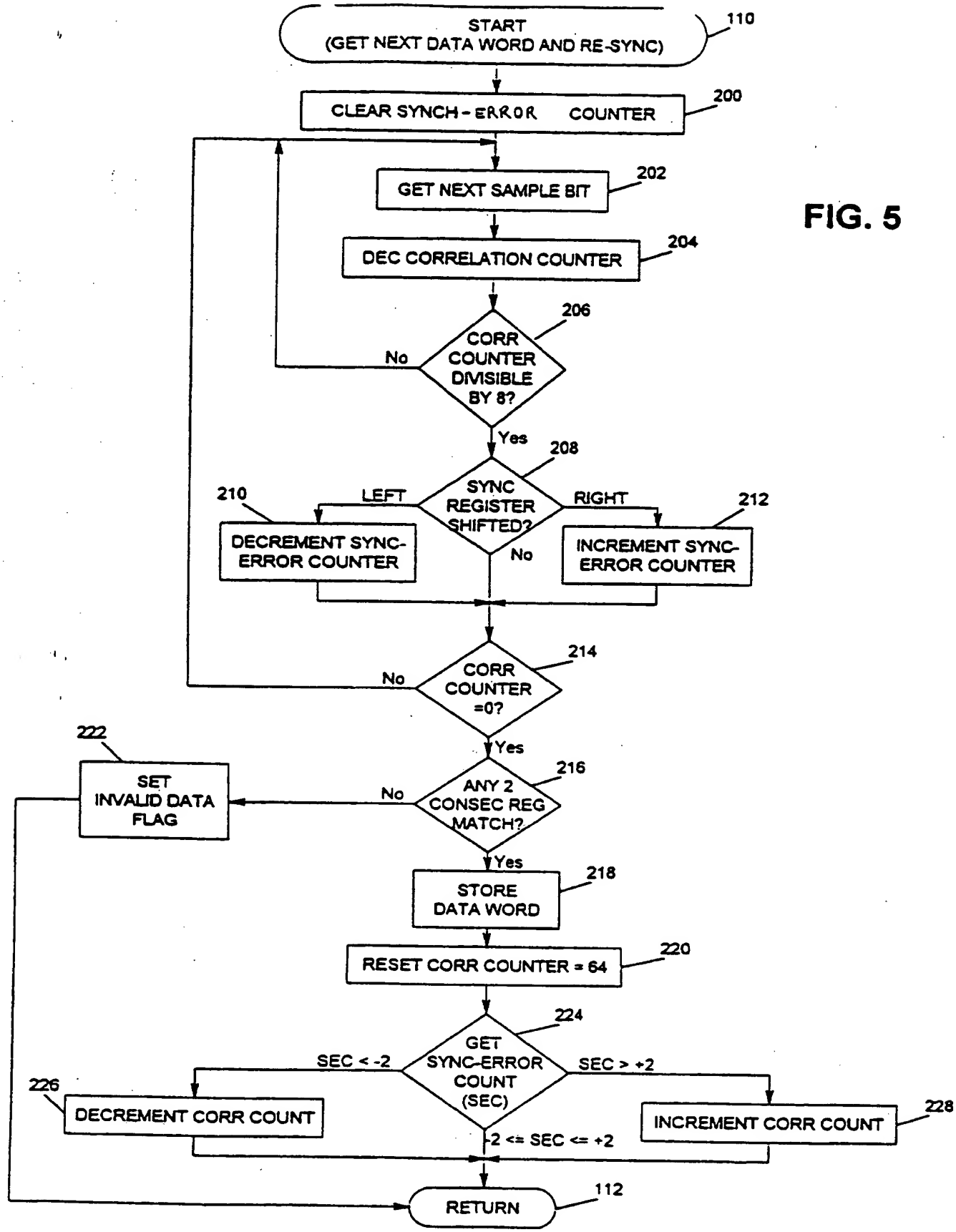

FIG. 5

CORDLESS TELEPHONE AND METHOD OF SYNCHRONIZATION FOR SECURE, HIGH-SPEED, HIGH-VOLUME DATA TRANSFER

This invention relates to cordless telephones, and more particularly to a cordless telephone which employs synchronisation of large burst data transmission.

It is quite common in the cordless telephone field to send digital information in data bursts. In traditional cordless telephone models where speech data is sent via analog transmission means, short digital data bursts are typically employed to transmit basic telephone commands from the telephone handset to the telephone base unit and vice versa. In digital cordless telephones, however, both command data and speech data are sent via short digital data bursts.

In most cordless telephones, the length of each data burst is limited in duration. Additionally, each data burst requires a relatively substantial amount of "overhead" as compared to the amount of information data actually transferred. For example, the data burst described in U.S. Patent No. 5,073,932 (Yossifor et al) is 24 bits in length (excluding the "synchronisation code" which is simply an alternating pattern of "1" and "0" used to stabilize the receiver hardware). Of these 24 bits, only five bits are used for command data, and the remaining bits are overhead.

Short data burst length and high overhead are not problematic for first generation cordless telephones due to the limited number of commands that need to be transferred between the telephone base unit and the telephone handset. However, in order to accommodate the increasing variety of services being offered by local telephone companies (e.g., Caller ID, Call-Waiting Caller ID, Caller ID Deluxe, Type 3 Telephones, etc.), it has become necessary to transmit larger streams of data from the base unit to the handset. For example, a cordless telephone user might want the ability to access Caller ID information through the handset either audibly through a speaker or visually via a text display. Moreover, these data streams often need to be transferred

very quickly. Simply cascading several short data bursts in sequence will often require too much time to convey the required information due to the high overhead.

Current methods for receiving data burst transmissions are incapable of receiving long data bursts (i.e. 25 bytes or more) and/or are relatively costly to implement. The primary reason for being incapable of receiving long data bursts regards the difficulties in maintaining proper data burst synchronisation between the transmitter and receiver.

In the prior art, synchronisation of the transmitter and receiver is typically established only at the beginning of a data burst. Since steps are not taken during the remainder of the burst to adjust the synchronisation of the signal, minor discrepancies between the transmitter and receiver clocks quickly accumulate, leading to a loss of synchronisation in the form of bit-slippage (gaining or losing at least 1 bit) at the receiver.

This occurs even with cordless telephones which transmit speech data digitally. U.S. Patent No. 5,434,905 (Maeda et al) provides an example of a digital cordless telephone which uses data bursts for both speech and data transmission. Since the synchronisation is established only at the beginning of the data burst, the only way to maintain synchronisation for any significant amount of time is to use crystal oscillators with extremely low tolerances in both the base unit and handset. In practice, this is a much too costly solution given the competitive nature of today's telecommunications market.

U.S. Patent No. 5,436,937 (Brown et al) discloses a rather complex, hardware intensive method of maintaining data burst synchronisation. Brown's invention is implemented using a multi-mode phase-locked loop circuit combined with an early/late bit transition accumulator. Although the multi-mode phase-locked loop circuit provides a way of maintaining synchronisation between the handset and base unit throughout a data burst, it is too costly for use in cordless telephones.

It is therefore a general object of the present invention to provide an improved cordless telephone.

It is a further object of the present invention to provide a secure means for transmitting relatively large data bursts between the base unit and the handset of a cordless telephone.

It is a further object of the present invention to provide a technique for transmitting a relatively large data burst between the base unit and the handset of a cordless telephone at a relatively high transmission rate.

It is a further object of the present invention to accomplish the foregoing objects at low cost.

It is still a further object of the present invention to overcome inherent disadvantages of known cordless telephones.

In accordance with a first aspect of the present invention, a cordless telephone unit comprises a base unit having a data generator for generating a digital data burst having periodic transitions, a data encoder coupled to the data generator for encoding the digital data burst to provide an encoded digital data burst, and a transmitter coupled to the data encoder for transmitting the encoded data burst; and a handset comprising a receiver responsive to the encoded data burst for receiving and sampling the encoded digital data burst; and a synchronisation device coupled to the receiver for maintaining synchronisation between the receiver and the transmitted encoded data burst by comparing transition periods in the transmitted encoded digital data burst.

In accordance with a second aspect of the present invention, a method of synchronisation between a base unit and a handset of a cordless telephone system, said method comprising the steps of: a) generating a digital data burst having a transition period in said base unit; b) encoding said digital data burst in said base unit; c) transmitting said encoded digital data burst from said base unit to said handset; d) receiving said encoded digital data burst in said handset; e) sampling said received encoded digital data burst in said handset; and f) synchronizing sampling and transmission of said encoded digital data burst by comparing transition periods in said transmitted encoded data burst.

The present invention also comprises a cordless telephone handset for use with the base unit defined in the first aspect of the present invention.

The present invention provides a very efficient technique of dynamically adjusting synchronisation throughout a data burst without the extensive use of hardware. This technique allows for much longer streams of data to be transferred without requiring the presence of ultra-precise oscillators in the base unit and the handset.

An additional improvement of the present invention is that the method of synchronisation uses information already present in the data stream; that is, no additional synchronisation bits need to be sent in the longer burst. This greatly reduces overhead as the data burst length increases and allows for a relatively large amount of data to be transferred in a short period of time.

The above and other objects, features and advantages of the invention will become readily apparent from the following detailed description of embodiments thereof which is to be read in connection with the accompanying drawings, in which:

FIG. 1 is a schematic representation of a data burst according to the present invention;

FIG. 2 is a block diagram showing the generation and transmission of a data burst within a cordless telephone base unit according to the present invention;

FIG. 3 is a block diagram showing the reception and analysis of a data burst within a cordless telephone handset according to the present invention;

FIG. 4 is a flow chart of a data burst analyzing process according to the present invention; and

FIG. 5 is a flow chart of a synchronisation process according to the present invention.

Referring now to FIG. 1, a data burst 2 is shown. Data burst 2 includes a preamble portion 4, a security code portion 6, a data sequence portion 8, and a checksum portion 10. As used herein, a word refers to 16 binary bits and a byte refers to 8 binary bits.

In the preferred embodiment of the invention, the preamble portion 4 includes a series of bits alternating at regular intervals between logic level "0" and logic level "1". The preamble portion 4 preferably does not contain any data information to be transferred. The purpose of the preamble portion 4, which is generated by the data

burst assembler 52 (shown in Fig. 2), is simply to prepare hardware contained in a receiver portion of a telephone handset 18 for the remainder of the incoming data burst. For example, there is a tendency for decision comparator 22 to remain in its current logic state, this tendency increasing over time, and thereby increasing the probability that a differing incoming signal will go undetected.

Referring now to Fig. 2, the security code portion 6 of data burst 2 is preferably generated in telephone base unit 40 by a free running counter 50 (described below). As shown in Fig. 2, the free running counter 50 has its output signal provided to filter 48 which removes certain undesired sequences from the free running counter output signal (for example, a "1" following many "0"s and vice-versa). Each time the telephone handset 18 (shown in FIG. 3) is placed in the cradle (not shown) of the base unit 40 (see FIG. 2), a new security code 6 is randomly generated by the free running counter 50 and provided to filter 48. The security code 6 is then provided to and stored in both the transmitter security code memory register 46 and the handset security code memory register 26. When the handset 18 is placed in the cradle of the base unit 40, the handset security code memory register 26 is operatively coupled to the transmitter security code memory register 46 to receive the randomly generated security code 6. In a preferred embodiment, the security code 6 is one word (16 bits) in length.

The data sequence portion 8 of data burst 2 is generated by data sequence generator 8' and is preferably encoded with a non-return to zero (NRZ) code by the base unit 40 before being transmitted by transmitter 54 to receiver 20 of the handset 18. In a preferred embodiment, the NRZ encoding of the data sequence portion 8 is provided by a Manchester encoder 44. NRZ encoding is a well known code form that has only two states commonly referred to as "zero" and "one". There is no neutral or rest condition in NRZ encoding. In other words, the Manchester encoder 44 replaces every logic level "0" bit with the two bits "01" and replaces every logic level "1" bit with the two bits "10". Therefore, for example, an eight bit sequence "1010 0011" which is encoded by the Manchester encoder 44 is transformed to the 16 bit sequence of "1001 1001 0101 1010". As known in the art, the spaces between each four bits of the 8 and 16 bit sequence have been added simply for visual clarity.

The checksum portion 10 of the data burst 2 is a byte representing the total "value" of the data sequence 8. In a preferred embodiment, the checksum 10 is generated by a checksum accumulator 42 located in base unit 40. The checksum accumulator 42 adds the binary values of all bytes in the data sequence 8 and provides a checksum accumulator output signal. The checksum accumulator output signal (corresponding to checksum 10) provides reference information to indicate whether all of the data sequence was accurately transmitted from the base unit 40 to the receiver 20 of handset 18. Specifically, as explained in more detail below, the checksum portion 10 of data burst 2 is compared against the value output by checksum accumulator 56 of handset 18 to determine if the transmission from the base unit 40 to the handset 18 is proper and complete. The checksum portion 10, in a preferred embodiment, is also NRZ encoded to 16 bits using the Manchester encoder 44.

In the preferred embodiment as shown in Fig. 2, the Manchester encoded data sequence portion 8 and checksum portion 10 are provided, along with security code portion 6, to data burst assembler 52. Data burst assembler 52 receives the data sequence portion 8, checksum portion 10 and security code portion 6, assembles the portions, including the preamble portion 4, in series and provides the assembled data burst 2 to a transmitter 54 for transmission to receiver 20 of handset 18.

The receiver 20 preferably samples incoming data signals at four times the rate at which the data burst bits are transmitted by transmitter 54. The samples detected by receiver 20 are provided to a decision comparator 22. The decision comparator 22 determines the value (either logic "1" or logic "0") of each sample, and provides the detected value to correlation register 24, correlation counter 30 and synchronisation register 36. By sampling each received signal several times, the decision comparator 22 is able to provide the correlation register 24 with redundant information which can be utilized at a later time to reduce the likelihood of a transmission/reception error.

The correlation register 24 preferably includes four shift registers (not shown) with the number of bit locations in each register being equal to the number of bits contained in security code 6. As samples (i.e., bits) are provided to correlation register 24 from decision comparator 22, the bits are provided to each shift register in turn so that the bit-zero locations of all four registers are filled before any one of the four shift

registers receives a second sample. Thus, each shift register receives a bit value every fourth sample. Each sample taken by the receiver is provided to only one shift register.

When the handset 18 is not performing any other task, it is always placed in a security code detection mode. Each time a bit is provided to and stored in one of the shift registers of correlation register 24, a register comparator 28 compares the contents of the shift registers of correlation register 24 to the known security code 6 stored in handset security code memory register 26. If any two consecutive shift registers of the correlation register 24 match every bit of the security code 6 stored in handset security code memory register 26, it is determined that a valid security code 6 has been received from the base unit 40.

In a preferred embodiment, each shift register (hereinafter called COR1, COR2, COR3, COR4) of correlation register 24 has a 16 bit capacity, corresponding to a 16 bit security code 6. However, larger or smaller registers are foreseen. Suppose, for example, that the transmitter 54 provides a data burst 2 and the security code 6 is 0101 0010 1101 0011. Assuming perfect synchronisation over an error-free channel, the decision comparator 22 will detect four identical samples of each bit of the security code 6. Thus, the first 20 samples to be detected (corresponding to the first five bits of the security code 6 stored in each of the shift registers of correlation register 24) would be 0000 1111 0000 1111 0000.

Assume that the first sample has just been detected (taken by receiver 20), and that the shift register presently available in the correlation register 24 is COR2. The first sample is provided to the first position of COR2 as follows:

BIT#	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHIFT REG.																
COR1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
COR2 →	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0
COR3	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
COR4	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

wherein: "X" represents old data that was present before the current sample arrived; and "O" and "1" represent the bit values of the current sample.

The second sample will be placed in the first position of the next shift register (COR3) of the correlation register 24 as follows:

BIT# / SHIFT REG.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COR1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
COR2	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0
COR3 →	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0
COR4	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

After 61 security code samples are detected by receiver 20, the shift registers (COR1, COR2, COR3, COR4) of correlation register 24 will contain the following data:

BIT# / SHIFT REG.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COR1	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1
COR2	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1
COR3	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1
COR4	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1

At this point, register comparator 28 will determine that COR2 matches the current security code 6 stored in handset security code memory register 26. After the next sample is detected by receiver 20, register comparator 28 will determine that COR3 also matches the security code 6 as shown in the following:

BIT# / SHIFT REG.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COR1	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1
COR2	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1
COR3	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1
COR4	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1

Case 1

Thus, after 62 samples, two consecutive shift registers (i.e., COR2 and COR3) of the correlation register 24 match the security code 6 of data burst 2. Therefore, register comparator 28 determines that a valid security code 6 has been received from the base unit 40 and is present in the correlation registers 24.

It should be noted that, if there had been errors (i.e., "E") in the data detected by COR2 (for example, error at sample 9 and 11 of COR 2), then a valid security code would not have been detected until matches occurred in COR3 and COR4 (one sample later) as shown below (i.e., after 63 samples):

BIT# / SHIFT REG.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COR1	X	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1
COR2	0	1	0	1	E	0	E	0	1	1	0	1	0	0	1	1
COR3	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1
COR4	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1

Case 2

If there had been errors in the data stored in COR2 (samples 9 an 11) and COR3 (sample 6) as shown below, then valid security code detection would not have occurred

until security code matches were found in COR4 and COR1 (one sample later) as shown below (after 64 samples):

BIT# / SHIFT REG.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COR1	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1
COR2	0	1	0	1	E	0	E	0	1	1	0	1	0	0	1	1
COR3	0	1	0	1	0	0	1	0	1	E	0	1	0	0	1	1
COR4	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1

Case 3

If the register comparator 28 fails to detect a valid security code 6 from base unit 40, (e.g. an error is present in at least three of the four shift registers COR1, COR2, COR3 and COR4), the handset 18 requests that the base unit 40 retransmit the data burst 2. The receiver 20 can signal the transmitter 54 that an error has occurred by either sending a negative acknowledgement (NAK) signal or by not sending an acknowledgement (ACK) signal. Such signals are common and well known in the art and performed by conventional circuitry. As the transmitter 54 expects an acknowledgement from the receiver 20 at specified times, both NAK and the absence of ACK signals serve to notify the transmitter 54 that the data burst needs to be retransmitted. The process of analyzing the security code 6 will then be restarted.

Once a valid security code 6 has been detected by register comparator 28, the handset 18 will enter a LOCK mode. In LOCK mode, the decision comparator 22 continues to provide (i.e., shift in) new data samples to correlation register 24. However, security code detection is no longer conducted. Instead, decision comparator 22 provides (i.e., shifts in) the data sequence portion 8 of the data burst 2 to the correlation register 24. The register comparator 28 analyzes the shift registers of the correlation register 24 to detect valid data words (i.e. data bytes which have been Manchester encoded into 16 bit data words).

In the incoming data stream of the encoded data burst 2, the first data word of the data sequence 8 immediately follows the security code 6. The correlation counter 30 counts the number of bits of the data burst 2 that arrive after a security code 6 has been detected. In this way, the correlation counter 30 can indicate when the correlation register 24 is completely filled and the first data word of the data sequence 8 should be present.

Since the sampling rate of the receiver 20 is four times faster than the transmission bit rate of transmitter 54, each bit provided by transmitter 54 is preferably detected four times. Since each data word is equal in length to the length of the security code 6, there will be one sample time during which the correlation register 24 contains four copies of the first data word. This sample time occurs at the moment the last security code bit is shifted out of the correlation register 24.

In the preferred embodiment, if detection of a valid security code occurs as described in Case 1 above, then the correlation register 24 will contain four copies of the first data word exactly 66 sample times later. This is because there are 4 x 16 = 64 bits in the correlation register 24 at the moment the valid security code 6 is detected, plus an additional two security code bits not yet clocked in. Similarly, if detection of a valid security code occurs as described in Case 2 above, then the correlation register 24 will contain four copies of the first data word exactly 65 sample times later. Finally, if detection of a valid security code occurs as described in Case 3 above, the four copies of the first data word will be present exactly 64 sample times later.

In reality, the handset 18 does not know whether detection of a valid security code occurred under Case 1, Case 2, or Case 3. The obvious choice would be to initialize the correlation counter 30 to the average value required to clock in four copies of the first data word. In a preferred embodiment, the correlation counter 30 is initialized to 65 (the average of 66, 65, and 64). Clearly, if detection of a valid security code were to occur as described in Case 1 or Case 3, after receiving 65 samples, the correlation register 24 would not contain the data exactly as desired. However, assuming no other errors in transmission, the data will be skewed in only one of the shift registers of the correlation register 24 and the other three shift registers of the correlation register 24 will contain the desired data.

Immediately after the 65 samples are provided (i.e., clocked in), the register comparator 28 is used to determine if any two consecutive shift registers of the correlation register 24 contain the same data word. If two consecutive shift registers contain the same data word, then it is assumed with a rather high degree of reliability that this was indeed the data sent by the transmitter 54.

The technique described thus far works sufficiently well for transmitting one word of data at a time. However, for transmitting long bursts of data, minor timing discrepancies between the transmitter 54 and receiver 20 can cause "bit-slippage," resulting in an occasional gain or loss of a sample of data. In addition, transmission noise may cause erroneous bits to be stored in the correlation register 24, compounding the problem.

Suppose, for example, that the security code 6 has been detected as set forth above in connection with Case 1. The correlation counter 30 is preferably set at 65, and 65 samples are clocked in an attempt to fill the correlation register 24 with the first word of data. Since 65 is not the optimal value for Case 1, the first data word would appear in at most three of the four shift registers in the correlation register 24. Without synchronisation, there would be no way to determine whether too many, too few, or the correct amount of bits have been shifted into the correlation register 24. In order to detect the second word, the only logical choice for the initial value of the correlation counter 30 is 64. However, without any means to make continual adjustments, as the length of the data sequence portion 8 is increased, the probability that the correlation register 24 would only contain three or even two copies of the correct word would quickly diminish with each new word received.

Therefore, in the preferred embodiment of the present invention, this problem can be solved by using a system which dynamically adjusts the correlation counter 30, and consequently the correlation register 24, by looking for "edges" in the received data. Specifically, as previously mentioned, as samples are shifted into the correlation register 24, they are concurrently shifted into synchronisation register 36. The synchronisation register 36 is a secondary shift register wherein the number of bit locations is preferably equal to half the number of bit locations in one of the shift registers (COR1, COR2, COR3, COR4) of the correlation register 24.

Recall that the data sequence 8 is Manchester encoded before being transmitted and therefore it will always be sent in bit-pairs of the form 01 or 10. After being sampled by the receiver 20 at four times the transmission bit-rate of transmitter 54, the bit-pairs will translate into bytes of the form 00001111 for bit-pair 01 or 11110000 for bit-pair 10. Provided that there are no errors, a single bit-pair will fill the entire synchronisation register 36 in one of the preceding forms. A bit-slip detector 34 which is operatively coupled to synchronisation register 36, compares the two center bits (i.e., bits 3 and 4, where bit 0 is located at the right most position) of the synchronisation register 36. If the two center bits differ (i.e., a "0" and "1", or "1" and "0" are detected), the bit-slip detector 34 determines that the transmission and reception of the data stream is in "sync". If the center bits match (i.e., two "0" or two "1", are detected), it is determined that a bit-slip may have occurred. If bit-slippage is detected, the bit-slip detector 34 then performs a "one-sided majority vote test" on the contents of the synchronisation register 36 to determine in which direction the slippage may have occurred.

The "one-sided majority vote test" is implemented as follows. Bits 2, 1, and 0 of the synchronisation register 36 are evaluated and a "vote" is taken to obtain a predominant bit value (either 0 or 1). The predominant bit value is then compared to bit 3 of the synchronisation register 36. If the predominant bit value matches the value of bit 3, it is determined that a shift to the left has occurred (i.e., too many bits have been shifted in) and the sync-error counter 32 is decremented by one. If the predominant value does not match the value of the third bit, then it is determined that the shift was to the right and the sync-error counter 32 is incremented by one. The term "one-sided" is derived from the fact that only bits 2, 1, and 0 are used in the vote and bits 7, 6, and 5 are ignored. This significantly decreases the amount of time required for evaluation and therefore more data may be processed.

When a valid data word has been detected by the register comparator 28 in any two consecutive shift registers (COR1, COR2, COR3, COR4) of the correlation register 24, the detected valid data word is provided by register comparator 28 to a data sequence memory register 38 and stored therein. It is at this step of the process that the correlation counter 30 is reset to 63, 64, or 65, depending upon the value of the sync-

error counter 32, according to a method such as the one to be described shortly. If a valid data word cannot be detected by the register comparator 28 in any two consecutive shift registers of the correlation register 24, it is determined that invalid data has been received. Thereafter, the handset 18 requests retransmission of the data burst 2. As the data sequence 8 is passed through the correlation register 24 and register comparator 28, a byte counter 62 which is electrically coupled to register comparator 28, counts the number of bytes in the transmitted data sequence 8. Substantially concurrently, handset checksum accumulator 56, which is also electrically coupled to register comparator 28, accumulates the binary values of the bytes. When the byte counter 62 reaches a predetermined value equal to the total number of bytes in the data sequence 8, checksum comparator 58 compares the value of the checksum accumulator 56 to the checksum 10 to verify that all of the data sequence 8 was transmitted. A data decoder and processor 60 decodes the data sequence 8 which has been stored in data sequence memory register 38 once the entire data sequence 8 has been received and verified by the checksum accumulator 56, byte counter 62 and checksum comparator 58. The processed data sequence 8 may be displayed on a text display 70, announced over a speaker, or otherwise output to the user.

Referring now to FIGS. 4 and 5, the method of the preferred embodiment of the invention will be described.

Referring initially to FIG. 4, upon initialization of cordless telephone system (START), the handset 18 is placed in the security code detection mode. Initially, a sample bit of data is received (Step 100) and the sample bit is provided to one of the shift registers (COR1, COR2, COR3, COR4) of correlation register 24 (Step 102). Then, the contents of one of the shift registers of correlation register 24 is compared to the known security code 6 stored in the handset security code memory register 26 (Step 104). If the contents of one of the shift registers does not match the contents of the handset security code memory register 26 (NO in Step 104), the method returns to Step 100. If the contents of one of the shift registers matches the contents of the handset security code memory register (YES in Step 104), the method proceeds to Step 106. More specifically, a determination is made as to whether there is a second consecutive match (Step 106). If there is not a second consecutive match of the contents of the shift

registers of correlation register 24 and handset security code memory register 26 (NO in Step 106), the method returns to Step 100. If there is the second consecutive match (YES in Step 106), this is indicative of detection of a valid security code and the method proceeds to Step 108. Thereafter, the correlation counter 30 is set to a value of 65 (Step 108). Then, a subroutine is initiated which corresponds to LOCK mode (Step 110), as shown in FIG. 5.

Referring now to FIG. 5, after the next data word is retrieved, the sync-error counter 32 is cleared (Step 200). Then, the next sample bit is shifted into the correlation register 24 as usual, but the same bit is also shifted into an 8 bit synchronisation register 36 (Step 202). Thereafter, the correlation counter 30 is decremented by one (Step 204). Then, the correlation counter 30 is analyzed to determine if the new value is divisible by eight (Step 206). If the correlation counter 30 is not divisible by eight (NO in Step 206), the method returns to Step 202. If the correlation counter 30 is divisible by eight (YES in Step 206), then a determination is made as to whether a bit slip has occurred (Step 208), and if it did occur, whether the slip was to the right or to the left. If a slip occurred to the left (LEFT in Step 208), the sync-error counter 32 is decremented by one (Step 210). If a slip occurred to the right (RIGHT in Step 208), the sync-error counter 32 is incremented by one (Step 212). If a bit-slip has not occurred (NO in Step 208), the sync-error counter 32 is not changed. Next, the correlation counter 30 is analyzed to determine if it is equal to zero (Step 214). If the correlation counter is equivalent to zero (YES in Step 214), the method proceeds to Step 216. However, if correlation counter 30 is not equal to zero, (NO in Step 214), then the method returns to Step 202 wherein the next sample bit is received.

At step 216, the consecutive shift registers of the correlation register 24 are compared to determine if two consecutive shift registers having matching entries. If a match is not found (NO in Step 216), it is determined that errant or invalid data was transmitted, and an invalid data flag is set (Step 222). The subroutine terminates and returns to the main method at step 112. If two consecutive shift registers have matching entries (YES in Step 216), then the data word is stored in the data sequence memory register 38 (Step 218). Thereafter, the correlation counter 30 is reset to a value of 64 (Step 220). Then, the value of the sync-error counter 32 (SEC) is analyzed to

determine if a shift has occurred (Step 224). If SEC < -2, the correlation counter 30 is decremented by one (Step 226). If SEC > +2, the correlation counter 30 is incremented by one (Step 228). If $-2 \leq SEC \leq +2$, the correlation counter 30 is unchanged. The subroutine then terminates and returns to the main method (Step 112).

Referring again to FIG. 4, a determination is made as to whether the invalid data flag has been set (Step 112). If the invalid data flag has been set (YES in Step 112), then the method ends. However, if the invalid data flag has not been set (NO in Step 112), a determination is made as to whether the current word is the last word of the data sequence 8 (Step 114). If the current word is not the last word of the data sequence (NO in Step 114), the method returns to Step 110. If the current word is the last word of the data sequence (YES in Step 114), the value of the handset checksum accumulator 56 is compared to the value of the checksum 10 (Step 116). If the values are equal (YES in Step 116), the data sequence 8 is decoded (Step 118). Step 118 decodes the data sequence 8 stored in the data sequence memory register 38 and proceeds to exit the method upon completion. If the values are not equal (NO in Step 116), the method terminates (END).

Although the invention has been illustrated and described in detail in the drawings and foregoing description, the same is to be considered as illustrative and not restrictive in character, it being understood that only preferred embodiments have been shown and described, and that all changes and modifications that come within the spirit and scope of the invention, as defined in the claims, are to be protected.

CLAIMS:

1. A cordless telephone unit comprising:

a base unit comprising:

a data generator for generating a digital data burst having periodic transitions;

a data encoder coupled to said data generator for encoding the digital data burst to provide an encoded digital data burst; and

a transmitter coupled to said data encoder for transmitting the encoded digital data burst; and

a handset comprising:

a receiver responsive to the encoded digital data burst for receiving and sampling the encoded digital data burst; and

a synchronisation device coupled to said receiver for maintaining synchronisation between the receiver and the transmitted encoded digital data bursts by comparing transition periods in the transmitted encoded digital data burst.

2. The cordless telephone unit as defined by claim 1, wherein said synchronisation device maintains synchronisation between the received and the transmitted encoded digital data burst in accordance with a characteristic of said data encoder.

3. The cordless telephone unit as defined by claim 1 or claim 2, wherein said data encoder encodes the digital data burst with a nonreturn to zero code.

4. The cordless telephone unit as defined by claim 1 or claim 2, wherein said data encoder comprises a Manchester encoder.

5. The cordless telephone unit as defined by any one of the claims 1 to 4, wherein the encoding provided by said data encoder includes an expected time of transition of an incoming signal.

6. The cordless telephone unit as defined by any one of the claims 1 to 5, wherein said synchronisation device maintains synchronisation between the received and the

transmitted encoded digital data burst by comparing an actual time of transition of the encoded data burst to an expected time of transition of the encoded data burst and based on the comparison, adjusting a sampling point of said transmitted encoded data burst.

7. The cordless telephone unit as defined by claim 6, wherein said synchronisation device determines adjustment of the sampling point by a majority vote of samples around the sampling point.

8. The cordless telephone unit as defined by claim 7, wherein said majority vote is a one-sided vote.

9. The cordless telephone unit as defined by claim 2, wherein said characteristic of said data encoder includes an expected time of transition of an incoming signal.

10. The cordless telephone unit as defined by claim 9, wherein said synchronisation device maintains said synchronisation between the received and the transmitted encoded digital data burst by comparing an actual time of transition of the encoded data burst to an expected time of transition of the encoded data burst and, based on the comparison, adjusting a sampling point of said transmitted encoded digital data burst.

11. The cordless telephone unit as defined by claim 10, wherein said synchronisation device determines adjustment of the sampling point by a majority vote of samples around the sampling point.

12. The cordless telephone unit as defined by claim 11, wherein said majority vote is a one-sided vote.

13. The cordless telephone unit as defined by any one of the claims 1 to 12, wherein said handset further comprises a text display.

14. The cordless telephone unit as defined by claim 13, wherein said text display of said handset displays caller identification information.

15. The cordless telephone unit as defined by any one of the claims 1 to 14, wherein the digitally encoded data burst includes caller identification information.

16. The cordless telephone unit as defined by any one of the claims 1 to 15, wherein the synchronisation device comprises:

a synchronisation register for storing the sampled encoded data burst; and

a bit-slip detector for comparing two bits of the sampled encoded data burst and determining if a bit slip has occurred and a direction of the bit slip.

17. The cordless telephone unit as defined by claim 16, further comprising:

a sync-error counter for one of decrementing and incrementing a count therein in dependence on whether a bit slip has been detected by said bit-slip detector and the direction of said bit slip.

18. The cordless telephone unit as defined by claim 17, further comprising:

a correlation counter for recording the number of bits that arrive after a security code in the received encoded data burst has been detected, said correlation counter being decremented or incremented in response to said sync-error counter and providing an indication as to a location of a first data word in said transmitted encoded data burst.

19. A method of synchronisation between a base unit and a handset of a cordless telephone system, said method comprising the steps of:

a) generating a digital data burst having a transition period in said base unit;

b) encoding said digital data burst in said base unit;

c) transmitting said encoded digital data burst from said base unit to said handset;

d) receiving said encoded digital data burst in said handset;

e) sampling said received encoded digital data burst in said handset; and

f) synchronizing sampling and transmission of said encoded digital data burst by comparing transition periods in said transmitted encoded data burst.

20. The method of synchronisation as defined by claim 19, wherein said step of synchronizing is performed in accordance with a characteristic of a data encoder which encodes said digital data burst.

21. The method of synchronisation as defined by claim 19 or claim 20, wherein said step of encoding includes the step of encoding said digital data burst with a nonreturn to zero code.

22. The method of synchronisation as defined by claim 21, wherein said nonreturn to zero code encoding comprises Manchester encoder.

23. The method of synchronisation as defined by any one of the claims 20 to 22, wherein said step of encoding includes an expected time of transition of an incoming signal.

24. The method of synchronisation as defined by claim 23, wherein said step of synchronizing includes the steps of:

comparing an actual time of transition to said expected time of transition; and

adjusting a sampling point of said transmitted encoded data burst.

25. The method of synchronisation as defined by claim 24, wherein said step of adjusting determines adjustment of the sampling point by a majority vote of samples around said sampling point.

26. The method of synchronisation as defined by claim 25, wherein said majority vote is a one-sided vote.

27. The method of synchronisation as defined by claim 20, wherein said step of encoding includes an expected time of transition of an incoming signal.

28. The method of synchronisation as defined by claim 27, wherein said step of synchronizing includes the steps of:

comparing an actual time of transition to said expected time of transition; and

adjusting a sampling point of said transmitted encoded data burst.

29. The method of synchronisation as defined by claim 28, wherein said step of adjusting determines adjustment of the sampling point by a majority vote of samples around said sampling point.

30. The method of synchronisation as defined by claim 29, wherein said majority vote is a one-sided vote.

31. The method of synchronisation as defined by any one of claims 19 to 30, further comprising the step of displaying text.

32. The method of synchronisation as defined by any one of the claims 19 to 31, wherein said encoded data burst includes caller identification information.

33. The method of synchronisation as defined by claim 32 when dependent on claim 31, wherein said step of displaying includes the step of displaying said caller identification information.

34. The method of synchronisation as defined by any one of claims 19 to 33, wherein the step of synchronizing includes the steps of:

receiving the sampled encoded digital data burst in a synchronisation register; and

comparing two bits stored in the synchronisation register to determine if a bit slip has occurred and a direction of said bit slip.

35. The method of synchronisation as defined by claim 34, further comprising the step of decrementing or incrementing a count in a sync-error counter in dependence on whether a bit slip has been determined and on the direction of said bit slip.

36. A method of synchronisation as defined by claim 35, further comprising the steps of:

maintaining in a correlation counter having a count, the number of bits that arrive after a security code in the received encoded digital data burst has been detected;

one of decrementing and incrementing the count in the correlation counter in response to said sync-error counter; and

providing an indication of when a first data word should be present based on the count in said correlation counter.

37. A cordless telephone handset for use with a base unit, which base unit comprises:

a data generator for generating a digital data burst having periodic transitions;

a data encoder coupled to said data generator for encoding the digital data burst to provide an encoded digital data burst; and

a transmitter coupled to said data encoder for transmitting the encoded digital data burst;

the cordless telephone handset comprising:

a receiver responsive to the encoded digital data burst for receiving and sampling the encoded digital data burst; and

a synchronisation device coupled to said receiver for maintaining synchronisation between the receiver and the transmitted encoded digital data bursts by comparing transition periods in the transmitted encoded digital data burst.

38. A cordless telephone unit substantially as herein described with reference to the accompanying drawings.

39. A method of synchronisation between a base unit and a handset of a cordless telephone system substantially as herein described with reference to the accompanying drawings.

40. A cordless telephone handset for use with a base unit substantially as herein described with reference to the accompanying drawings, the cordless telephone handset being substantially as herein described with reference to the accompanying drawings.

The Patent Office

24

Application No:	GB 9805691.4	**Examiner:**	Gareth Griffiths
Claims searched:	All	**Date of search:**	28 August 1998

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.P): H4L (LDJJ, LDLT)

Int Cl (Ed.6): H04B 7/212, 7/26, H04L 7/00, 7/02, 7/033, 7/08, 7/10, H04M 1/72

Other: Online Database: WPI

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
A	US5434905 (MAEDA)	

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Majority Decoded Automatic Repeat Request Error Control Schemes[1]

Cheng-shong Wu and Victor O.K. Li
Department of Electrical Engineering
University of Southern California
Los Angeles, CA 90089-0272

Abstract

We propose a simple method to improve the performance of Automatic Repeat Request (ARQ) error control. To minimize the hardware and software overhead, we do not use sophisticated coding method. Instead, when a transmission error occurs, the erroneous packet, instead of being discarded as in traditional ARQ, are kept for majority decoding later. This method is easy to implement in existing ARQ systems without much modification. In this paper, we apply the method to various ARQ's, including ideal selective repeat and go back N under point-to-point and point-to-multipoint communication environments. We also study the effect of transmitting multiple copies of a packet. The analysis shows that higher throughput is obtained when bit error rate is high or when the number of receivers is large.

1 Introduction

There are two fundamental classes of error control techniques in data communication systems: the automatic repeat request (ARQ) scheme and the forward error correction (FEC) scheme. ARQ is often preferred over FEC because it is relatively simple. Some hybrid ARQ schemes have also been proposed [1,4,6,7,8,13]. The ARQ protocols are sometimes classified into three major categories: 1) stop and wait, 2) go back N, and 3) selective repeat. Stop and wait protocols tend to be inefficient in terms of throughput, especially over high-delay channels such as satellite channels. Go back N (GBN) suffers a low throughput when link error rate is high. Various modified GBN protocols have been proposed in which multiple copies of data packets are transmitted to get a better throughput at high error rate [2,11,14]. Selective repeat (SR) protocol offers better throughput than GBN but requires additional complexity at the receiver and the transmitter. Furthermore, ideal selective repeat requires infinite buffers for receivers and infinite sequence number for packets. Some finite-buffer selective repeat protocols have been proposed [10,17,18]. Among them, Weldon's scheme [3,17] transmits multiple copies of data and is simple and efficient. In this paper, we introduce a method to increase the throughput of ARQ systems without sophisticated coding and without degrading the throughput at low error rate. We call this majority-decoded ARQ. This method is easy to implement in almost any existing ARQ systems. Recently, point-to-multipoint ARQ has been studied extensively [5,9,12,16,15]. Therefore, we also extend our work to point-to-multipoint systems. Hybrid ARQ using code combining [1,4,6,7] seems to have better performance then us. However, their scheme is much more complex. Furthermore [1,4,6,7] consider ideal point-to-point SR ARQ only. Section 2 describes the environment and the proposed majority decoded error control scheme. In section 3, we apply the method to SR, GBN, redundant GBN and finite buffer SR. We derive analytical expressions for the throughput of each of these protocols. Some numerical examples are given and discussed in section 4. We conclude in section 5.

[1] This research is supported in part by Army Research Office under contract No DAAG29-85-K-0116

2 System model

2.1 System architecture

The system we consider consists of a transmitter connected to each receiver by a data link. All data is transmitted in the form of packets and time is divided into fixed length interval called slots. The length of each slot is equal to the time required to transmit a single packet and each packet has the following embedded in it:

1. A sequence number which identifies the packet and its position in the sequence of data being transmitted.
2. A cyclic redundancy code (CRC) which enables the receiver to detect transmission errors.

The transmitter expects the receiver to confirm every packet received correctly. There is a separate feedback channel on which acknowledgments from receivers are returned to the transmitter. The receiver sends either positive acknowledgement if the packet is successfully decoded or negative acknowledgmeht if the packet cannot be decoded correctly.

2.2 Majority decoding

Our proposed majority-decoded ARQ works as follows. If the receiver does not get a correct reception from the first copy of a given packet, a negative acknowledgment is returned to the transmitter and a second transmission is required. If the receiver still cannot get a correct reception, the third copy is needed and so on.

In order to apply majority decoding, we need the last three copies of the same packet. The first and second transmission of a packet is handled in exactly the same way as in traditional ARQ. If a copy of a packet is detected in error after the second trial, majority decoding is applied.

In majority decoding, the receiver compares the three copies bit by bit, performs a majority vote at each bit, and then uses the CRC to check whether the resultant packet is correct or not. If it is correct, a positive acknowledgment is sent; otherwise another transmission is needed. It is important that majority decoding is applied to three copies with the same sequence number. To ensure this, we may use a CRC on the sequence number, in addition to the CRC for the whole packet.

2.3 Different ARQ protocols

Majority decoding can be applied to existing ARQ protocols. First, we consider ideal selective repeat and go back N. In ideal selective repeat, the receiver is assumed to have infinite buffer to resequence the arriving packets.

In go back N, the receiver provides one buffer for error detection and an extra two buffers for majority decoding if it is being used. If a packet cannot be detected correctly, the receiver must discard other incoming packets until the packet is successfully received in order to preserve the original order of packets.

Go back N becomes inefficient when the error rate is high. In order to improve the throughput at high error rate, we transmit multiple copies [2]. We call it *redundant* go back N scheme. In this scheme, we transmit multiple copies of a packet instead of one at a time. The number of copies to be transmitted every time

CH2525-3/88/0000-1648 $1.00 © 1988 IEEE

IEEE Global Telecommunications Conference & Exhibition

Hollywood, Florida • November 28 - December 1, 1988

"Communications for the Information Age"

Conference Record

Volume 3 of 3

Volume	Day	Sessions	Pages
1	Tuesday	1 - 18	1 - 594
2	Wednesday	19 - 36	601 - 1187
3	Thursday	37 - 54	1193 - 1817

Sponsored by the IEEE Communications Society
and the
Palm Beach and Broward County IEEE Sections

IEEE IEEE ICC GLOBECOM

is chosen to optimize the system throughput.

Ideal selective repeat is impractical because it needs infinite buffer for the receiver and infinite sequence number for every packet. In this paper we will adapt Weldon's finite buffer protocol.

3 Performance analysis

In this section, we will analyze the throughput of the proposed majority-decoded ARQ system as well as the classical ARQ system. Various types of ARQ protocols will be considered.

3.1 Definitions and assumptions

First, we define the following terms:

- p: bit error rate. We assume binary symmetric channels and independent bit error rate.
- l: packet length in bits.
- q: packet error rate. This is the probability a single packet is in error.
- S: round-trip delay in slots, i.e., the time from the transmission of a packet until an acknowledgment is returned.
- X: random number describing the number of transmissions required to transmit a given packet until the receiver receives it correctly.
- Y: total number of transmissions due to a given packet (including retransmissions of this packet and other packets discarded due to retransmissions of this packet).
- $F_X(i)$: the cumulative distribution function (cdf) of random variable X, that is, $F_X(i) = \text{prob}\{X \le i\}$.
- $\overline{F}_X(i)$: $\overline{F}_X(i) = 1 - F_X(i)$
- η: throughput of the system, defined as the average number of packets delivered to the receiver successfully per transmission.

From the above definitions, Y is the total number of transmissions due to a given packet. It is clear that $\eta = \frac{1}{\mathrm{E}[Y]}$.

3.1.1 The distribution of X

Since the cdf of r.v. X plays a very important role in our analysis, we shall first try to find it.

In classical, non-majority-decoded ARQ (NARQ), no matter how many times a packet has been transmitted, the receiver always has probability $1 - q$ to get a correct reception in each transmission. Thus, X is geometrically distributed with cumulative distribution function $F_x(i) = 1 - q^i$, $i = 1, 2, \ldots, \infty$.

In majority-decoded ARQ (MARQ), some erroneous copies are kept for further use. Majority decoding cannot be applied to the first and second trials of a packet since we need three copies. If a packet is detected in error after the second trial, majority decoding is applied to this and the two previous erroneous copies. The result of majority decoding will be correct if the bit error positions of these three copies do not overlap.

For $X \le 2$, the cdf of X for MARQ is the same as that for NARQ. For $X \ge 3$, we can get a difference equation

$$\overline{F}_X(i) = (1 - \frac{r_3}{r_2})q\overline{F}_X(i-1) - (\frac{r_3}{r_2} - r_3)q^2\overline{F}_X(i-2), \quad i \ge 3 \qquad (1)$$

(1) is derived in the Appendix. Solving (1), we get

$$F_X(i) = \begin{cases} 0 & i \le 0 \\ 1 - q & i = 1 \\ 1 - q^2 & i = 2 \\ 1 - As_1^{i-3} - Bs_2^{i-3} & i \ge 3 \end{cases} \qquad (2)$$

where

$$s_1 = \frac{q}{2}(1 - \frac{r_3}{r_2} - (1 - \frac{r_3}{r_2})^2 - 4r_3)^{\frac{1}{2}})$$

$$s_2 = \frac{q}{2}(1 - \frac{r_3}{r_2} - ((1 + \frac{r_3}{r_2})^2 - 4r_3)^{\frac{1}{2}})$$

are the roots of the characteristic polynomial of (1) and

$$A = \frac{q^3}{2}(1 - r_3 + \frac{(1 - 3r_3 + \frac{r_3}{r_2} + \frac{r_3^2}{r_2})((1 + \frac{r_3}{r_2})^2 - 4r_3)^{\frac{1}{2}}}{(1 + \frac{r_3}{r_2})^2 - 4r_3})$$

$$B = \frac{q^3}{2}(1 - r_3 - \frac{(1 - 3r_3 + \frac{r_3}{r_2} + \frac{r_3^2}{r_2})((1 + \frac{r_3}{r_2})^2 - 4r_3)^{\frac{1}{2}}}{(1 + \frac{r_3}{r_2})^2 - 4r_3})$$

are determined by substituting (2) into (1). Note that

$$r_2 = \frac{1}{q^2}(1 - p)^l\{(1 + p)^l - 2 + (1 - p)^l\}$$

$$r_3 = \frac{1}{q^3}(1 - p)^{2l}\{(1 + 2p)^l - 3(1 + p)^l + 3 - (1 - p)^l\}$$

are the probabilities that none of the bit error positions overlap in 2 and 3 independent erroneous copies, respectively.

Finally, we find the expectation of X

$$E[X] = \sum_{i=1}^{\infty} i f_X(i) = \sum_{i=0}^{\infty} \overline{F}_X(i) = \begin{cases} \frac{1}{1-q} & \text{for NARQ} \\ 1 + q + q^2 + \frac{A}{1-s_1} + \frac{B}{1-s_2} & \text{for MARQ} \end{cases} \qquad (3)$$

3.2 Ideal selective repeat (SR) and classical go back N (GBN) ARQ

For ideal selective repeat, the receiver provides infinite buffer to store packets. Since no buffer overflow will occur, the retransmission of an erroneous packet does not cause the loss of other packets. Random variable X is equal to r.v. Y. Therefore,

$$\eta_{SR} = \frac{1}{\mathrm{E}[X]} \qquad (4)$$

$\mathrm{E}[X]$ can be found from (3) for either MARQ or NARQ.

For classical go back N (NGBN), the receiver provides only one buffer, while three buffers are required in majority-decodeed go back N (MGBN) to store the erroneous copies for majority decoding. If a packet (re)transmission is unsuccessful, retransmission is required, each of which will cause the loss of $S - 1$ packets. Thus Y is related to X by $Y = X + (X - 1)(S - 1)$ and

$$\eta_{GBN} = \frac{1}{\mathrm{E}[Y]} = \frac{\eta_{SR}}{(1 - \eta_{SR})(S - 1) + 1} \qquad (5)$$

3.3 Redundant go back N (RGBN)

Under RGBN, instead of transmitting one copy of a packet each time, the sender transmits n copies of a packet. As in GBN protocol, the receiver provides one buffer in non-majority-decoded redundant go back N (NRGBN) and provides three buffers for majority-decoded redundant go back N (MRGBN).

Under this redundant retransmission strategy, when a new packet β is to be sent, the transmitter repeats β n times. If the receiver fails to receive β correctly in these n copies, a negative acknowledgment is sent and the transmitter resends n copies of β again. This is repeated until the receiver receives this packet successfully. With probability $F_X(n)$ the receiver can get a successful packet in the first set of transmissions. Hence, the probability β succeeds in the first n trials is

$$\text{prob}(Y = n) = F_X(n)$$

As in classical go back N, if the sender cannot get a correct packet from the first set of n transmissions, then the second set of n trials is needed, causing a loss of $S-1$ packets. The probability of success at the ith set of trials is $F_X(in) - F_X((i-1)n)$. Therefore, we have

$$\text{prob}(Y = in-(i-1)(S-1)) = F_X(in) - F_X((i-1)n), \quad i \geq 3 \quad (6)$$

After substituting $F_X(n)$ and taking the expectation of Y, we obtain

$$E[Y(n)] = \frac{S-1-n}{1-q^n} - S + 1, \quad \text{for NRGBN} \quad (7)$$

and for MARQ we have
$E[Y(n)] =$

$$\begin{cases} (n+S-1)(1-q-q^2+\frac{A}{1-s_1}+\frac{B}{1-s_2}) - S + 1 & n=1 \\ (n+S-1)(1-q^2-\frac{As_1}{1-s_1}+\frac{Bs_2}{1-s_2}) - S + 1 & n=2 \\ (n+S-1)(1-\frac{As_1^{n-2}}{1-s_1}+\frac{Bs_2^{n-2}}{1-s_2}) - S + 1 & n \geq 3 \end{cases} \quad (8)$$

$E[Y(n)]$ depends not only on the bit error rate p but also on the round-trip delay S. For any p and S, we can get an optimal value of n, $\hat{n}$, which minimizes $E[Y(n)]$.

The optimal throughput of the redundant strategy for both majority-decoded and non-majority decoded cases is

$$\eta_{FS} = \frac{1}{E[Y(\hat{n})]} \quad (9)$$

3.4 Finite-buffer selective repeat (FSR) strategies

Various finite-buffer selective repeat ARQ strategies have been proposed. Among them, Weldon's scheme [17] has the highest throughput over a wide range of bit error rates. Therefore, we will modify this scheme with our majority-decoded method.

In Weldon's scheme, the receiver buffer is of size αS where α is a positive integer. To simplify the problem, we will only consider the case $\alpha = 1$, that is, the receiver has S buffers. The result can be readily extended to $\alpha \geq 1$.

A packet β is transmitted according to the following procedure in Weldon's scheme in a system with S buffers.

- *level* 0: β is repeated n_0 times for the first set of transmissions. If an ACK is received (S packet time later), the transmission of β is complete. Otherwise, move to level 1.
- *level* 1: At this point the receiver buffer is full. β is repeated n_1 times. If an ACK is received , the transmission of β is complete. Otherwise, move to level 2.
- *level* 2: Buffer overflow occurs, leading to a loss of $S-1$ packets. β is repeated n_1 times and stays at this level until it is successfully received.

With probability $F_X(n_0)$, β will succeed at its first set of transmissions. If the receiver cannot get a correct packet from first n_0 copies then an additional n_1 will be sent. Hence, we have

$$\text{prob}(Y = n_0) = F_X(n_0) \quad (10)$$

$$\text{prob}(Y = n_0 - n_1) = F_X(n_0 + n_1) - F_X(n_0)$$

If the receiver still cannot get a correct packet, the system moves to level 2. The receiver buffer overflows, $S-1$ packets are discarded and n_1 additional copies of β are sent. This procedure will be repeated until the packet is successfully received. Thus, we get

$$\text{prob}(Y = n_0 - in_1 - (i-1)(S-1)) = F_X(n_0 - in_1) - F_X(n_0 - (i-1)n_1), \quad i \geq 1 \quad (11)$$

Substituting (3) into (11) and taking expectation, we get

$$E[Y] = n_0 - n_1 + \left(\frac{n_1 - S - 1}{1-q^{n_1}} - S - 1\right) \quad \text{for NFSR} \quad (12)$$

and
$E[Y] =$

$$\begin{cases} 1+(S+1)(q+q^2+\frac{A}{1-s_1}+\frac{B}{1-s_2})-(S-1)q & n_0=1, n_1=1 \\ 1+(S+1)(q+\frac{As_1^{n_1-1}}{1-s_1}+\frac{Bs_2^{n_1-1}}{1-s_2})-(S-1)q & n_0=1, n_1 \geq 2 \\ 2+(S+2)(q^2+\frac{As_1^{n_1-1}}{1-s_1}+\frac{Bs_2^{n_1-1}}{1-s_2})-(S-1)q^2 & n_0=2, n_1 \geq 1 \\ n_0+(S+n_0)(\frac{As_1^{n_0-2}}{1-s_1}+\frac{Bs_2^{n_0-2}}{1-s_2}) & \\ \quad -(S-1)(As_1^{n_0-3}+Bs_2^{n_0-3}) & n_0 \geq 3, n_1 \geq 1 \end{cases} \quad (13)$$

for MFSR

Finally the maximum throughput of finite buffer selective repeat ARQ for both majority and non-majority decoded systems can be obtain by minimizing $E[Y]$.

$$\eta_{FSR} = \frac{1}{\min_{n_0,n_1} E[Y]} \quad (14)$$

3.5 Point-to-multipoint ARQ

We now study point-to-multipoint ARQ systems. In such systems, one transmitter sends data to many receivers. Assume that we have m receivers.

Let r.v. X_i, $i = 1, \cdots, m$ be the number of transmissions required to transmit a given packet until receiver i receives it correctly, and X_a be the number of transmissions until *all* receivers receive a packet correctly. Then, we have

$$X_a = max(X_1, \cdots, X_m)$$

If we also assume the error probability for each receiver is identical and independent, then

$$F_{X_a}(x) = (F_{X_i}(x))^m \quad (15)$$

For non-majority decoded ARQ, $F_{X_i}(x) = 1 - q^x$, $x = 1, \cdots, \infty$ and for majority-decoded ARQ, $F_{X_i}(x)$ is given by (2).

Substituting (15) into (4),(5),(6), (10) and (11), we can get the expectation of random variable Y and then the corresponding throughput of point-to-multipoint ARQ.

4 Numerical examples

Figures 1 and 2 compare different protocols. We assume $N = 100$ and $l = 1000$. The throughput of NGBN and MGBN are almost the same, although MGBN has a slight advantage at high error rate. The corresponding redundant protocols give higher throughput for P greater then 10^{-5}. As error probability increases, MRGBN gives the highest throughput. In fact, MRGBN has a throughput even higher than that of ideal non-majority decoded selective repeat at very high bit error rate. Majority decoding also gives higher throughput for SR strategies, and the improvement is most pronounced for high error rate.

Figure 2 compares finite-buffer and infinite-buffer (or ideal) SR. Again, finite buffer SR with majority decoding performs better than that without majority decoding. The throughput of finite-buffer and infinite-buffer SR are nearly the same (MFSR and MSR: NFSR and NSR) under Weldon's schemes with optimal n_0 and n_1. Generally speaking, the difference between them is small when majority decoding is used. We also note that, even with infinite buffer, NSR does not give a better performance then MFSR when bit error rate is high.

In figure 3, we compare the throughput of different protocols for different number of receivers at $p = 10^{-4}$. The throughput of any protocol decreases as the number of receivers increases. MSR and NSR have almost the same throughput at this bit error rate but MFSR is better then NFSR. Comparing MRGBN and NRGBN, the throughput of NRGBN drops much more quickly than that of MRGBN . The throughput of classical GBN, both majority-decoded and non-majority decoded are very low at this error probability.

5 Conclusions

In this paper we propose a simple method to improve the throughput of ARQ systems. Instead of discarding incorrectly received packets, we save them for majority decoding. We apply this method to several ARQ protocols, and to point-to-point and point-to-multipoint systems. Although we do not use sophisticated coding, we show that majority decoding will improve the performance of ARQ protocols. This improvement is most dramatic when the bit error rate is high.

Appendix : The distribution of r.v. X for majority-decoded ARQ

Define the following symbols:

- e_i: the event that the ith copy of a given packet is in error.
- $o(k, k\text{-}1, k\text{-}2)$: the event that the result of majority decoding from the kth, (k-1)th and (k-2)th copies of a given packet is not correct, that is, there are overlapping bit error positions.
- $no(k, k\text{-}1, k\text{-}2)$: the complementary event of $o(k, k\text{-}1, k\text{-}2)$.
- E_i: the event that the receiver fails to get a correct packet after the ith transmission, that is, random variable $X > i$.
- r_2: the probability that none of the bit error positions overlap in 2 independent erroneous copies.
- r_3: the probability that none of the bit error positions overlap in 3 independent erroneous copies.
- $\overline{F}_X(i)$: The probability random varible $X > i$.

From the above definition, we have

$$\begin{aligned} r_2 &= \text{prob}\{no(k,k-1)|e_k, e_{k-1}\} \qquad \text{(A.1)}\\ &= \text{prob}\{no(2,1)|e_2, e_1\}\\ &= \frac{\text{prob}\{no(2,1), e_2, e_1\}}{\text{prob}\{e_2, e_1\}}\\ &= (\frac{1}{q})^2 \sum_{i=1}^{l}\sum_{j=1}^{l} \binom{l}{i}\binom{l}{j} p^{i+j}(1-p)^{2l-i-j} \frac{\binom{l-i}{j}}{\binom{l}{j}}\\ &= \frac{1}{q^2}(1-p)^l((1+p)^l-2+(1-p)^l) \end{aligned}$$

and

$$\begin{aligned} r_3 &= \text{prob}\{no(k,k-1,k-2)|e_k, e_{k-1}, e_{k-2}\} \qquad \text{(A.2)}\\ &= \text{prob}\{no(3,2,1)|e_1, e_2, e_3\}\\ &= (\frac{1}{q})^3 \sum_{i=1}^{l}\sum_{j=1}^{l}\sum_{k=1}^{l} \binom{l}{i}\binom{l}{j}\binom{l}{k} p^{i+j+k}\\ &\quad (1-p)^{3l-i-j-k} \frac{\binom{l-i}{j}\binom{l-i-j}{k}}{\binom{l}{j}\binom{l}{k}}\\ &= \frac{1}{q^3}(1-p)^{2l}((1+2p)^l-3(1+p)^l+3-(1-p)^l) \end{aligned}$$

Since majority decoding cannot be applied to the first and second transmissions of a packet, the receiver cannot get a correct packet at the first and second trials if they are in error. After the second trial, the receiver can get a correct packet unless the transmission is in error and the result of majority decoding also results in error. Thus

$$\begin{aligned} E_1 &= e_1\\ E_2 &= e_2 \cap E_1\\ &= e_2 \cap e_1\\ E_3 &= o(3,2,1) \cap e_3 \cap E_2\\ &\vdots\\ E_i &= o(i, i\text{-}1, i\text{-}2) \cap e_i \cap E_{i-1}\\ &\vdots \end{aligned}$$

From the definition of $\overline{F}_X(i)$ and the assumption of independent errors for different copies of a packet, we have

$$\begin{aligned} \overline{F}_X(1) &= \text{prob}\{E_1\} = \text{prob}\{e_1\} \qquad \text{(A.3)}\\ &= q\\ \overline{F}_X(2) &= \text{prob}\{E_2\} = \text{prob}\{e_1, e_2\} \qquad \text{(A.4)}\\ &= q^2\\ \overline{F}_X(3) &= \text{prob}\{E_3\} = \text{prob}\{o(i,i\text{-}1,i\text{-}2), E_2, e_3\}\\ &= \text{prob}\{o(i,i\text{-}1,i\text{-}2)|e_1, e_2, e_3\} \times\\ &\quad \text{prob}\{e_1, e_2, e_3\}\\ &= q^3(1-r_3)\\ &\vdots\\ \overline{F}_X(i) &= \text{prob}\{E_i\} = \text{prob}\{o(i,i\text{-}1,i\text{-}2), E_{i-1}, e_i\}\\ &= \text{prob}\{o(i,i\text{-}1,i\text{-}2), o(i\text{-}1,i\text{-}2,i\text{-}3),\\ &\quad E_{i-2}, e_{i-1}, e_i\}\\ &= \text{prob}\{o(i,i\text{-}1,i\text{-}2), E_{i-2}, e_{i-1}, e_i\}\\ &\quad -\text{prob}\{o(i,i\text{-}1,i\text{-}2), no((i\text{-}1,i\text{-}2,i\text{-}3)\\ &\quad , E_{i-2}, e_{i-1}, e_i\}\\ &= \text{prob}\{o(i,i\text{-}1,i\text{-}2), E_{i-2}, e_{i-1}, e_i, e_{i-2}\} \qquad \text{(A.5)}\\ &\quad -\text{prob}\{o(i,i\text{-}1,i\text{-}2), no(i\text{-}1,i\text{-}2,i\text{-}3),\\ &\quad no(i\text{-}1,i\text{-}2), E_{i-2}, e_{i-1}, e_i, e_{i-2}\}\\ &= \text{prob}\{o(i,i\text{-}1,i\text{-}2)|E_{i-2}, e_{i-2}, e_{i-1}, e_i\} \qquad \text{(A.6)}\\ &\quad \times\text{prob}\{E_{i-2}, e_{i-1}, e_i\}\\ &\quad -\text{prob}\{o(i,i\text{-}1,i\text{-}2)|no(i\text{-}1,i\text{-}2),\\ &\quad no(i\text{-}1,i\text{-}2,i\text{-}3), E_{i-2}, e_{i-2}, e_{i-1}, e_i\} \times\\ &\quad \text{prob}\{no(i\text{-}1,i\text{-}2,i\text{-}3), E_{i-2}, e_{i-1}, e_i, e_{i-2}\} \end{aligned}$$

The additional terms e_{i-2} and $no(i-1, i-2)$ in (A.5) does not change the value of the expression because $e_{i-2} \supset E_{i-2}$ and $no(i\text{-}1, i-2) \supset no(i-1, i-2, i-3)$.

Since the event $o(i, i\text{-}1, i\text{-}2)$ is independent of the event E_{i-2} conditioned on e_i, e_{i-1} and e_{i-2}, the term before the minus sign in (A.6) becomes

$$\begin{aligned} &\text{prob}\{o(i,i\text{-}1,i\text{-}2)|e_i, e_{i-1}, e_{i-2}\}\\ &\quad \times\text{prob}\{E_{i-2}, e_i, e_{i-1}\} \qquad \text{(A.7)}\\ &\quad = (1-r_3)q^2\overline{F}_X(i\text{-}2) \end{aligned}$$

Similarly, the event $o(i, i\text{-}1, i\text{-}2)$ is independent of the event $no(i\text{-}1, i\text{-}2, i\text{-}3)$ and E_{i-2} conditioned on $no(i\text{-}1, i\text{-}2)$. Therefore, the term after the minus sign in (A.6) becomes

$$\begin{aligned} &\text{prob}\{o(i,i\text{-}1,i\text{-}2)|no(i\text{-}1,i\text{-}2), e_i, e_{i-1}, e_{i-2}\}\\ &\quad \times\text{prob}\{no(i\text{-}1,i\text{-}2,i\text{-}3), E_{i-2}, e_i, e_{i-1}\}\\ &= (1-\text{prob}\{no(i,i\text{-}1,i\text{-}2)|no(i\text{-}1,i\text{-}2), e_i, e_{i-1},\\ &\quad e_{i-2}\}) \times (\text{prob}\{E_{i-2}, e_{i-1}, e_i\} - \text{prob}\{ \qquad \text{(A.8)}\\ &\quad o(i\text{-}1,i\text{-}2,i\text{-}3), E_{i-2}, e_i, e_{i-1}\})\\ &= (1-\frac{r_3}{r_2})(q^2\overline{F}_X(i\text{-}2)-q\overline{F}_X(i\text{-}1)) \end{aligned}$$

From (A.6),(A.7),(A.8), we get

$$\overline{F}_X(i) = (1-\frac{r_3}{r_2})q\overline{F}_X(i\text{-}1) + (\frac{r_3}{r_2}-r_3)q^2\overline{F}_X(i\text{-}2), \quad i \geq 3 \qquad \text{(A.9)}$$

and the initial conditions

$$\overline{F}_X(3) = q^3(1-r_3)$$

$$\overline{F}_X(4) = q^4(1-2r_3+\frac{r_3^2}{r_2})$$

References

[1] G. Benelli. An ARQ scheme with memory and soft error dectors. *IEEE Trans. on Commun.*, 33, March 1985.

[2] Herwig Bruneel and Marc Moeneclaey. On the throughput of some continuous ARQ strategies with repeated transmission. *IEEE Trans. on Commun.*, COM-34:244–249, March 1986.

[3] Y. Chang and C. Leung. On Weldon's ARQ strategy. *IEEE Tran. Commun.*, COM-26(3):297–300, March 1984.

[4] D. Chase. Code combining – a maximun-likelihood decoding approach for combining an arbitrary number of noisy packets. *IEEE Trans. on Commun.*, 33, May 1985.

[5] I. Gopal and J. Jaffe. Point-to-multipoint communication over broadcast link. *IEEE Trans. on Commun.*, COM-32(9):1034–1044, Sept. 1984.

[6] H. Krishna and S.D. Morgera. A new error control scheme for hybrid ARQ systems. *IEEE Trans. on Commun.*, 35, October 1987.

[7] C. Lau and C. Leung. Performance analysis of a memory ARQ scheme with soft decision detectors. *IEEE Trans. on Commun.*, 34, August 1986.

[8] Shu Lin and Philip S. Yu. An hybrid ARQ scheme with paritey retransmission for error control of satellite channel. *IEEE Trans. on Commun.*, COM-30(7):1701–1719, July 1982.

[9] K. Mase, et. al. Go-back-N ARQ schemes for point to multipoint satellite communications. *IEEE Trans. on Commun.*, COM-32(4):583–589, April 1984.

[10] M. J. Miller and S. Lin. The anslysis of some selective-repeat ARQ schems with, finite receiver buffer. *IEEE Trans. on Commun.*, COM-29(9):1307–1315, Sept 1981.

[11] J. M. Morris. On another go-back N ARQ technique for high error rate conditions. *IEEE Trans. on Commun.*, COM-26:187–189. Jan 1978.

[12] Krishan Sabnani and Mischa Schwartz. Multidestination protocol for satellite broadcast channel. *IEEE Trans. on Commun.*, COM-33(3):232–240, March 1985.

[13] A. R. K. Sastry. Performance of hybrid error control schemes on satellite channel. *IEEE Trans. on Commun.*, COM-23(6):689–694, July 1975.

[14] A. R. K. Sastry and Laveen N. Kanal. Hybrid error control using retransmission and generalized burst-trapping code. *IEEE Trans. on Commun.*, COM-24(4):385–393, April 1986.

[15] Don Towsley. An analysis of point-to-multipoint channel using go-back-n error control protocol. *IEEE Trans. on Commun.*, COM-33(3):282–285, March 1985.

[16] Don Towsley. A selective repeat ARQ protocol for a point to multipoint channel. In *IEEE INFOCOM*, pages 521–526, 1987.

[17] E. J. Weldon, Jr. An improved selective-repeat ARQ strategy. *IEEE Trans. on Commun.*, COM-30(3):480–486, March 1982.

[18] Philip S. Yu and Shu Lin. An efficient selective-repeat ARQ for satellite channel and its throughput analysis. *IEEE Trans. on Commun.*, COM-29:353–363, March 1981.

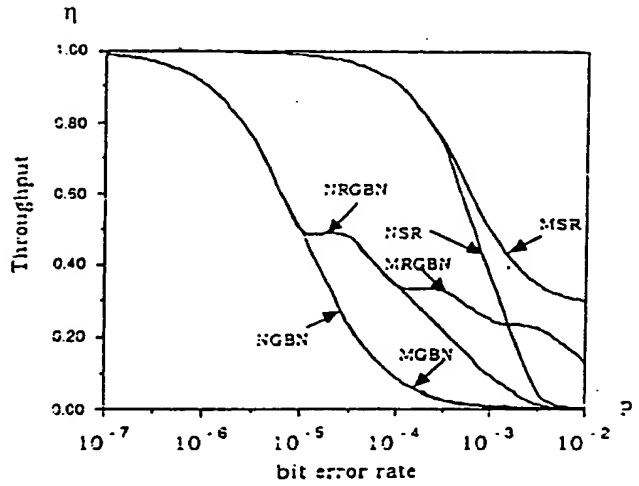

Figure 1 Throughput comparison for different ARQ protocols

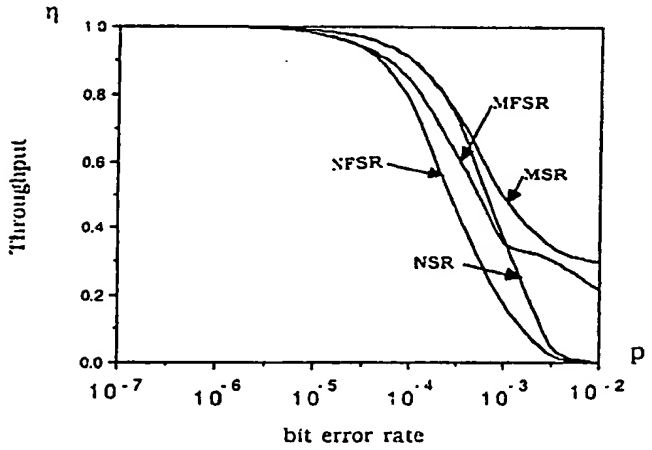

Figure 2. Throughput of finite and infinite buffer selective repeat ARQ

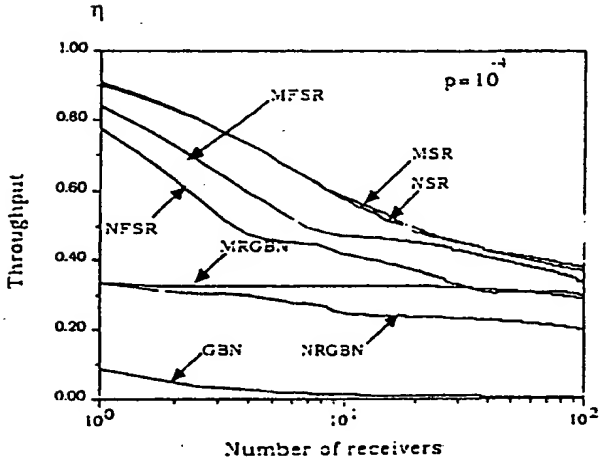

Figure 3. Throughput versus the number of receivers for different ARQ protocols